The Monarch Benchmark: Evaluating AI Performance in African and Emerging Market Applications

Africa Compute Fund

March 5, 2025

Abstract

The Monarch Benchmark establishes a rigorous, detailed standard for evaluating artificial intelligence systems specifically tailored to African contexts, with methodologies adaptable to other emerging regions. It addresses significant gaps in linguistic representation, sector-specific use cases, and constraints posed by limited computational infrastructure. This benchmark integrates comprehensive datasets, structured evaluation protocols, and precise performance metrics across critical domains: finance, legal analysis, healthcare diagnostics, agricultural forecasting, educational technologies, and software development. Explicit implementation guidelines emphasize reproducibility, resource-efficient model deployment, quantization strategies, and robust open-source collaboration protocols to ensure accessibility, transparency, and widespread adoption among researchers, developers, policymakers, and industry stakeholders.

1 Introduction

Despite advancements in AI, contemporary benchmarks such as GLUE, SuperGLUE, and MLPerf predominantly rely on Western datasets and cloud-based computational resources, leading to underperformance and misalignment when applied to African and other underrepresented markets. The Monarch Benchmark systematically resolves these shortcomings by integrating datasets reflective of local linguistic, economic, regulatory, and infrastructural realities, thus providing precise evaluation criteria directly relevant to emerging market scenarios. It emphasizes deployment feasibility on edge-computing platforms, low-power GPUs, and resource-constrained devices prevalent in such regions, thus addressing the practical constraints faced by developers and end-users in these contexts.

2 Motivation and Objectives

Current AI benchmarking efforts suffer from inadequate representation of languages, contexts, and resource constraints typical of African countries, exacerbating digital divides. Monarch directly confronts these issues by providing clearly defined evaluation objectives:

- **Comprehensive Linguistic Coverage:** Datasets include major African languages (Swahili, Hausa, Yoruba, Amharic, Zulu), dialectal variations, and multilingual code-switching phenomena prevalent in spoken and written communication.
- **Infrastructure-Aware Evaluation:** Explicitly designed to benchmark model performance on devices like Raspberry Pi 4, NVIDIA Jetson Nano, and other cost-effective edge hardware, simulating real-world deployment constraints (power consumption, inference latency, model quantization).
- **Sector-Specific Applicability:** Explicit datasets and tasks reflect real-world use cases across finance, regulatory compliance, healthcare diagnostics, agricultural monitoring, education systems, and localized software development practices.

- **Open-Source Standardization:** Clear and reproducible methodologies accompanied by publicly accessible evaluation scripts, containerized environments, and detailed documentation to encourage widespread adoption, community contributions, and collaborative research.

3 Benchmark Methodology

3.1 Implementation Guidelines

Evaluations under Monarch Benchmark utilize standardized, reproducible Docker container environments configured explicitly to mimic realistic deployment scenarios. All containers include clearly defined dependencies (TensorFlow, PyTorch, HuggingFace Transformers, ONNX runtime), Python versions, CUDA libraries (where applicable), and hardware acceleration configurations.

Edge deployment scenarios explicitly test AI model inference using resource-constrained platforms:

- **Low-Power GPUs:** NVIDIA Jetson Nano (128-core Maxwell GPU, 4GB RAM) benchmarking CUDA-accelerated inference using TensorRT-optimized models.
- **Single-board Computers (SBCs):** Raspberry Pi 4 (Broadcom BCM2711 Quad-core Cortex-A72, 8GB RAM), measuring inference performance using quantized model formats (e.g., TensorFlow Lite, ONNX quantization at 8-bit and 4-bit precision levels).

Evaluations systematically measure:

- **Inference latency:** Measured per inference batch and averaged over multiple runs with statistical variance reported.
- **Throughput:** Reported as inference per second under sustained workloads.
- **Memory footprint:** Peak and steady-state RAM consumption recorded using profiling tools such as Valgrind and NVIDIA Nsight Systems.
- **Energy efficiency:** Power usage quantified via USB inline power meters or integrated device power monitoring interfaces.

3.2 Structured Evaluation Protocol

The evaluation protocol explicitly defines reproducibility through dataset versioning, container environments, deterministic random seeds, standardized preprocessing and post-processing scripts, and controlled testing procedures. Domain-specific metrics include but are not limited to:

- **Natural Language Processing (NLP):**
  - **Accuracy**, macro-averaged and micro-averaged **F1-score**, **Precision**, **Recall** for classification tasks (NER, intent detection).
  - Translation quality using **BLEU**, **METEOR**, and **COMET** scores.
  - Text generation quality measured through **ROUGE-L**, **Perplexity**, and human-evaluation alignment scores.
- **Financial and Legal Domain:**
  - **Document classification accuracy**, summarization quality (ROUGE), and regulatory compliance detection precision.
  - API integration accuracy, measured via successful interaction rates, error classification, and anomaly detection rates.
- **Healthcare Domain:**

  - Diagnostic model accuracy, sensitivity, specificity, and ROC/AUC metrics.
  - Conversational medical support evaluated through dialogue success rates and user-satisfaction scoring methods.
- **Agricultural Domain:**
  - Regression metrics (RMSE, MAE) for yield prediction, classification accuracy for crop type and disease identification, and predictive accuracy (forecast RMSE) for weather-based models.
- **Educational Technologies:**
  - Content generation assessed via multilingual readability, translation coherence, and expert-evaluated pedagogical quality.
  - Tutoring systems evaluated using session length, learning outcome improvements, and adaptive content relevance metrics.
- **Code Generation and Integration:**
  - Functional correctness (percentage of compilable/executable code), successful API integrations, and unit-test pass rates.
  - Efficiency metrics including code runtime performance and resource consumption on edge hardware.

Each evaluation domain provides clearly documented reproducibility criteria, including required hardware, software, environmental conditions, and explicit metric calculation methodologies. Results are logged in structured JSON format to facilitate transparent comparisons across evaluations.

3.3 Compute Efficiency Evaluation

- **Model Quantization:** Evaluates the trade-off between performance and compression for INT8, 4-bit, and LoRA fine-tuned models.
- **Hardware Deployment:** Measures inference latency, throughput, and memory usage on low-power devices such as NVIDIA Jetson Nano, Raspberry Pi 4, and CPU-only environments.
- **Power Consumption:** Captures energy per inference using USB inline power meters and integrated power monitoring tools.

4 Datasets and Evaluations

4.1 Natural Language Processing

Proposed datasets for NLP evaluations include the MasakhaNER dataset, comprising approximately 50,000 annotated sentences across diverse African languages such as Swahili, Hausa, Yoruba, Amharic, and Zulu. For conversational AI tasks, the Lacuna Fund's multilingual dialogue corpus, with over 100,000 conversation samples demonstrating prevalent code-switching between local languages and English, is recommended. Additionally, parallel corpora for English–Swahili (approx. 200,000 sentence pairs) and French–Wolof (approx. 100,000 sentence pairs) translations are proposed. Evaluation tasks will focus on multilingual Named Entity Recognition (NER), intent detection accuracy, and machine translation quality assessed through standard metrics such as BLEU, METEOR, ROUGE-L, and perplexity, alongside domain-specific metrics.

4.2 Finance and Legal AI

For financial and legal domain evaluations, we propose synthetic transaction datasets aligned with popular mobile money APIs (e.g., M-Pesa, Paystack, Flutterwave) containing simulated transaction logs for anomaly detection and fraud classification scenarios. Additionally, regulatory texts such as Central Bank guidelines and financial reports from African countries (e.g., Kenya, Nigeria, South Africa) are recommended for summarization and compliance classification tasks. African legal archives, such as court judgments available through platforms like AfricanLII, are suggested for document classification, summarization, and contract analysis benchmarks. Metrics will include classification accuracy, F1-score for document categorization, anomaly detection precision/recall rates, and summarization quality evaluated by ROUGE and human assessment.

4.3 Healthcare

Suggested healthcare evaluation datasets include publicly available medical datasets, such as diabetic retinopathy image collections (e.g., Kaggle EyePACS with approximately 35,000 labeled images), and multilingual clinical notes datasets that mimic doctor–patient interactions across major African languages. Symptom-checker conversational datasets, designed specifically for low-resource contexts, are also proposed. Benchmarking tasks will involve image-based diagnostics classification accuracy, sensitivity, specificity, conversational triage decision-making efficacy, and multilingual medical question-answering performance assessed via F1-score, sensitivity, specificity, and ROC/AUC metrics.

4.4 Agriculture

For agriculture, satellite imagery datasets like MODIS or Sentinel-2 data archives are proposed for crop yield prediction (yield regression tasks) and crop-type classification. A collection of annotated images for plant disease recognition (e.g., PlantVillage with 54,000 labeled images across various diseases) is recommended for benchmarking disease detection accuracy. Localized weather datasets, such as CHIRPS rainfall datasets and agricultural sensor data capturing anomalies, can be utilized for forecasting accuracy evaluation (forecast RMSE) and sensor-based anomaly detection. Partnering up with agricultural startups and initiatives could also provide valuable unique datasets.

4.5 Education

Proposed datasets include multilingual educational question-and-answer pairs tailored for African curricula, potentially covering core subjects such as mathematics, science, and literature in languages like Swahili, Hausa, and Yoruba. Adaptive tutoring dialogue datasets are recommended, reflecting interactions between students and AI-powered tutoring agents. Additionally, automated essay-grading datasets comprising multilingual student essays and human-scored assessments are proposed. Evaluation metrics will emphasize accuracy of automated scoring, multilingual content generation coherence, readability scores, adaptive tutoring effectiveness metrics, and expert-evaluated pedagogical quality.

4.6 Code Generation

Proposed datasets in code generation include simulated API integration challenges with African fintech services (M-Pesa, Paystack, Flutterwave APIs), mobile-first software development scenarios (Android/USSD frameworks common in African mobile banking ecosystems), and programming exercises aligned with typical regional software development needs. Evaluation tasks will measure correctness of generated code (percentage passing automated functional tests), runtime efficiency (latency benchmarks), resource consumption on low-cost hardware, and code readability as evaluated by local developers.

4.7 Compute Efficiency

Given the prevalent constraints in African computational environments, compute-efficiency evaluations will target quantized models (e.g., QLoRA, LoRA, 4-bit precision inference). Recommended evaluation plat-

forms include Raspberry Pi 4 (1.5GHz CPU, 4GB RAM) and NVIDIA Jetson Nano (128-core GPU, 4GB RAM). Evaluation protocols specify precise measurement of inference latency (milliseconds per inference), throughput (inferences per second), peak memory footprint (in megabytes), and power efficiency (energy consumed per inference cycle). Statistical significance is ensured via repeated inference cycles with clearly documented measurement methods.

5 Conclusion and Future Work

The Monarch Benchmark introduces an essential foundation for evaluating AI models in African contexts, emphasizing local relevance, computational efficiency, and reproducibility. Proposed next steps include developing detailed dataset annotation guidelines, performing initial baseline evaluations using representative AI models (e.g., AfriBERT, AfroXLMR, quantized variants), and refining metrics through community input. Continued development will involve integrating feedback, expanding sector coverage, and ensuring alignment with the evolving needs of AI research communities in Africa and other emerging regions.

6 Benchmark Execution Protocol

To ensure reproducibility and consistency across evaluations, the Monarch Benchmark proposes a rigorous and standardized execution protocol, leveraging Docker containerization, clearly defined evaluation scripts, and structured result logging.

6.1 Containerized Evaluation Environment

Benchmark evaluations will be conducted within standardized Docker environments, ensuring exact reproducibility across various hardware setups. A sample Dockerfile is provided, specifying exact dependencies, frameworks, and computational libraries:

```
FROM python:3.11-slim
RUN apt-get update && apt-get install -y build-essential \
    && pip install torch torchvision torchaudio \
    transformers datasets onnxruntime \
    tensorflow==2.14 numpy pandas
COPY evaluate.py /benchmark/evaluate.py
ENTRYPOINT ["python", "/benchmark/evaluate.py"]
```

Docker environments will be pre-configured and distributed via Docker Hub, along with explicit instructions to replicate testing setups. Docker environments are versioned, enabling precise experiment replication (Docker Hub registry: `https://hub.docker.com/monarch-benchmark`).

6.2 Example Benchmark Execution

The following illustrates a standardized execution command for running a proposed evaluation task (Named Entity Recognition) using the MasakhaNER dataset on a hypothetical baseline model (`africabert`):

```
docker run -v $(pwd)/datasets:/datasets monarch-benchmark:latest \
    python evaluate.py \
    --task masakhaner \
    --model africabert \
    --precision 8bit \
    --hardware jetson_nano \
    --batch-size 16 \
    --output results/africabert_masakhaner.json
```

Key parameters include:

- `--task`: Evaluation scenario, clearly documented with task-specific guidelines and expected output formats.
- `--model`: Pretrained or fine-tuned model identifier, with recommended baseline models for initial benchmarks (e.g., AfriBERT [**?**], AfroXLMR [**?**]).
- `--output`: Path to store standardized evaluation metrics in JSON format.

6.3 Structured Result Schema

Evaluation results will be stored in a structured JSON format, clearly documenting detailed metrics, metadata, hardware specifications, and reproducibility parameters. An improved example schema:

```
{
    "evaluation_metadata": {
        "task": "masakhaner",
        "language": "yoruba",
        "dataset_version": "v1.0",
        "evaluation_date": "2025-03-05",
        "random_seed": 42
    },
    "model_details": {
        "name": "africabert",
        "version": "v1.2",
        "model_size_MB": 512,
        "quantization": "8-bit"
    },
    "hardware_environment": {
        "device": "NVIDIA Jetson Nano",
        "gpu_memory_GB": 4,
        "cpu_cores": 4,
        "power_consumption_W": 10.2
    },
    "metrics": {
        "accuracy": 0.93,
        "precision": 0.88,
        "recall": 0.90,
        "f1_score": 0.89,
        "inference_latency_ms": {
            "mean": 120,
            "std_dev": 5,
            "min": 115,
            "max": 127
        },
        "throughput_samples_per_sec": 8.3,
        "energy_consumption_Joules_per_inference": 2.4
    },
    "reproducibility_parameters": {
        "random_seed": 42,
        "batch_size": 16,
        "docker_image_version": "monarch-benchmark:v1.0"
    }
}
```

7 Proposed References and Resources

To ensure thoroughness and accuracy, the following references and resources are recommended for developing the benchmark:

- **MasakhaNER Project:** Adelani et al., "MasakhaNER: Named Entity Recognition for African Languages," *Transactions of ACL*, 2021. (`https://github.com/masakhane-io/masakhane-ner`)
- **AfriBERT:** Ogueji et al., "AfriBERTa: Pre-training Transformer Models for African Languages," *EMNLP*, 2021. (`https://github.com/castorini/afriberta`)
- **AfroXLM-R:** Alabi et al., "Adapting Multilingual Models for African Languages: AfroXLM-R," *EMNLP*, 2022. (`https://huggingface.co/Davlan/afroxlmr-base`)
- **PlantVillage Dataset:** Hughes and Salathé, "PlantVillage Dataset," openly available on Kaggle (`https://www.kaggle.com/datasets/emmarex/plantvillage`)
- **EyePACS Diabetic Retinopathy Dataset:** Available on Kaggle (`https://www.kaggle.com/c/diabetic-retinopathy-detection`)
- **MODIS Agricultural Dataset:** NASA MODIS data accessible via Radiant Earth Foundation (`https://www.radiant.earth/data/modis`)
- **CHIRPS Weather Dataset:** Rainfall data from the Climate Hazards Group InfraRed Precipitation with Stations (`https://www.chc.ucsb.edu/data/chirps`)
- **TensorRT and Quantization Documentation:** NVIDIA Developer resources for efficient inference (`https://developer.nvidia.com/tensorrt`)
- **QLoRA Quantization Method:** Dettmers et al., "QLoRA: Efficient Finetuning of Quantized Models," *arXiv preprint arXiv:2305.14314*, 2023. (`https://arxiv.org/abs/2305.14314`)

8 Standardization and Open Source Initiative Details

The Monarch Benchmark initiative will operate as a transparent, version-controlled open-source project hosted on GitHub. It will include:

- Comprehensive contribution guidelines and peer-review processes.
- Structured semantic versioning for datasets and evaluation scripts.
- Docker-based evaluation environments with explicitly versioned containers distributed through Docker Hub.
- Open governance through a steering committee representing academia, industry stakeholders, and local AI communities across Africa, ensuring accountability, responsiveness, and continual benchmark relevance.

Regular workshops, community meetings, and online forums will facilitate stakeholder feedback, ensuring continuous improvement aligned with emerging needs.

9 Future Development Roadmap

Proposed immediate next steps include:

- Publishing detailed annotation guidelines for dataset creation across each sector.
- Developing baseline implementations and evaluations on foundational models (e.g., AfriBERT, AfroXLM-R, quantized model benchmarks).

- Soliciting early feedback from regional stakeholders (universities, governments, local developers) to refine evaluation protocols.

Further developments will encompass iterative enhancements, new sector incorporation (energy, climate modeling), and cross-regional collaborative benchmarking, progressively positioning Monarch as a globally recognized standard for AI evaluation in emerging markets.